VŠB TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Absolvování individuální odborné praxe

Individual Professional Practice in a Company

Vojtěch Ermis

Bakalářská práce

Vedoucí práce: Ing. Tomáš Pavelek, Ph.D.

Ostrava, 2021

Abstrakt

Tato bakalářská práce popisuje mé působení na odborné praxi ve společnosti Roper Engineering s.r.o. Ve vypracování začínám popisem zaměření firmy a mého pracovního zařazení. Dále se zaměřuji převážně na postup při řešení některých zadaných úkolů. Při praxi mi bylo přiřazeno několik dílčích úkolů ve více projektech.

Během odborné praxe jsem pracoval ve vývojovém týmu. Pracoval jsem na návrhu a konstrukci testerů, které byly využity pro testy komponentů či celých systémů. Dále jsem působil jako podpora hardwarových vývojářů, kterým jsem pomáhal s rozmanitou škálou úkolů.

Klíčová slova

Odborná praxe, Roper Engineering s.r.o, Struers GmbH, průtokové snímače, AD převodník, Raspberry Pi, přerušení, TMCM-3110, TMCL, krokové motory

Abstract

This bachelor's thesis describes my practice work in the Roper Engineering s.r.o. In the elaboration I describe specialization of the company and my job. In this thesis, I mainly focus on the process of the solving of some assigned tasks. During the practice, I was assigned several sub-tasks in the multiple projects.

During my professional practice I worked in the development team. Most of my work was filled with the design and construction of testers, which were used to test components and entire systems. I also supported hardware developers, helping them with a diverse range of tasks.

Key words

Professional practice, Roper Engineering s.r.o, Struers GmbH, flow sensor, AD converter, Raspberry Pi, interrupt, TMCM-3110, TMCL, stepper motor

Poděkování

Touto cestou bych chtěl poděkovat zejména Ing. Richardu Szuscikovi, Ing. Jiřímu Friedeckému a celému R&D týmu z firmy Roper Engineering za cenné odborné rady.

Dále bych rád poděkoval vedoucímu práce Ing. Tomáši Pavelkovi, Ph.D. za asistenci a konzultace při vytvoření této bakalářské práce.

Obsah

Seznam ilustrací

Seznam použitých zkratek

Zkratka	Význam
R & D	Research and Development – vývoj a výzkum
SPI	Serial Peripheral Interface – sériové periferní rozhraní
I2C	Inter-Integrated Circuit – sběrnice
UART	Universal Asynchronous Receiver-Transmitter – asynchronní sériový přenos, rozhraní
AD	Analog Digital – analogově digitální
CS	Chip Select – řidicí linka sběrnice pro výběr zařízení
DIN	Digital Input – digitální vstup
DOUT	Digital Output – digitální výstup
CLK	Clock – synchronizační hodinový signál
MS	Microsoft
CSV	Comma separated values – hodnoty odděleny čárkami
GUI	Graphical user interface – grafické uživatelské rozhraní
TMCL	Trinamic Motion Control Language
USB	Universal Serial Bus – univerzální sériová sběrnice
RS485	Standard sériové komunikace
CAN	Controller Area Network – sběrnice
RPi	Raspberry Pi – jednodeskový počítač
EEPROM	Electrically Erasable Programmable Read-Only Memory – elektricky přemazatelná programovatelná paměť
PC	Program Counter – čítač instrukcí

Úvod

Tato bakalářská práce popisuje mé absolvování odborné praxe ve společnosti Roper Engineering s.r.o. Během praxe jsem se věnoval především vývoji a sestavování testeru, které slouží k ověřování funkčnosti komponentů či jejich dlouhodobém tzv. lifetime testu.

Firma Roper Engineering s.r.o, sídlící v Ostravě, je vývojovým a produkčním centrem firmy Struers. Roper Engineering společně s firmou Struers jsou vlastněny americkou společností Roper Technologies. Hlavním zaměřením je produkce zařízení, sloužících k přípravě vzorku pro metalografii a analýze materiálů. Dále firma poskytuje kompletní sortiment spotřebního materiálu, který je optimalizován pro nabízené portfolio.

Po dobu praxe jsem byl součástí R&D týmu, který se společně s vývojovým centrem v Dánsku podílí na vývoji produktů. Jedná se především o manuální, poloautomatické či plně automatické řezačky, brousící či leštící stroje a další laboratorní příslušenství. Tým je složen z mnoha odborníků z různých oborů, ať už softwarových, mechanických či hardwarových inženýrů. Osobně jsem byl součástí skupiny, která se stará o vývoj hardware.

1 Popis odborného zaměření firmy a pracovního zařazení

1.1 Historie firmy Struers

Holger Struer založil Struers roku 1875 v hlavním městě Dánska, Kodani. V počátcích byla firma převážně zaměřena na import chemikálií a laboratorních zařízení. Za první světové války byl zastaven veškerý dovoz a Struers vyrobil svůj první produkt, jednalo se o teploměr. Roku 1919 získala firma zastoupení rakouské firmy Reichert, která byla specialistou v metalografických mikroskopech. Odtud vzplanul zájem o metalografickou analýzu.

Roku 1943 firma Struers na popud manažera materiálové laboratoře firmy DISA vyvine laboratorní zařízení pro elektrolytické leštění. Jedná se o přístroj s názvem MicroPol, který byl kompletně vyvinut i produkován Struers. Rok 1943 lze označit za velký krok směrem k metalurgii vzhledem k uvedení MicroPolu na trh. V pozdějších letech dochází k expanzi firmy a většinovému přechodu firmy k produkci laboratorních přístrojů pro metalografii. Na začátku tohoto období se jedná hlavně o brusky a leštičky. Roku 1961 firma expanduje a vzniká dceřiná společnost v Düsseldorfu.

Roku 1963 se filozofie firmy změní a hlavním cílem je poskytnout zákazníkovi kompletní produktový program pro přípravu metalografických vzorků. Dalším cílem bylo, aby přístroje byly dokonalejší než standartní zařízení, které se na trhu nacházely, a aby byly plně přizpůsobeny zákazníkům.

Roku 1965 dojde k uvedení první řezačky na trh, nese název Discotom. Společně s ní je vydán i stroj pro zapouzdření vzorků Prestopress.

V roce 1973 si firma uvědomila důležitost automatické přípravy a s postupem let vyvinula jako první na světě automatickou řezačku s názvem Magnutom. Roku 1983 došlo i k vyvinutí světově prvního metalografického zařízení s mikroprocesorem, konkrétně brusky/leštičky Abramatic. Automatické řízení u tohoto stroje optimalizovalo brusný proces a dávkovalo suspenzi. Později došlo k dalšímu rozmachu automatických strojů.

Při postupném přechodu firmy k producentovi metalografických přístrojů se původně chemická firma rozdělila roku 1989 na dvě části Struers Tech a Struers Chem a byly založeny další dceřiné společnosti ať už v Británii, Francii, Rakousku, Švédsku či Japonsku.

Později došlo k rozšiřování portfolia o například Vickersovy tvrdoměry, automatické čističky i programové vybavení pro analýzu vzorků.

Aktuálně je firma přítomna s kancelářemi a přidruženými společnostmi ve 24 zemích. Ve více než 50 zemích po celém světě lze Struers označit za předního světového dodavatele metalografických řešení. [1]

1.2 Pracovní zařazení ve firmě

Ve společnosti pracuji na pozici s názvem R & D Trainee. Náplní je se podílet na vývoji zařízení, převážně na jejich elektrické části. Při procesu vývoje a návrhu nového přístroje je třeba, aby se průběžně realizovaly zkoušky komponentů a systému, aby došlo k minimalizaci poruch a komplikací při vydání zařízení na trh. Samotný vývoj se rozděluje na několik fází. V každé z nich se vykonává řada ověření funkčnosti a konceptů. Při těchto testovacích a ověřovacích pracích je často nutné, aby bylo sestrojeno zařízení, které bude nějakým způsobem interagovat se zkoušeným předmětem.

Několik úkolů, které mi byly zadány, se zaměřovaly právě na konstrukci těchto testerů dle formulovaných požadavků. Podle nich jsem musel navrhnout kompletní řešení. V některých případech se jednalo o komplexnější testery, jak svou mechanickou konstrukcí, řízením nebo i elektrickým zapojením. V některých případech bylo řízení uskutečněno jednodeskovými počítači či mikrokontrolery, u jednodušších systémů byla dostačující aplikace reléové logiky. U jednodeskových počítačů jsem byl zodpovědný i za tvorbu programu v programovacím jazyku Python nebo C++. Při výkonu práce byla nutná úzká spolupráce s kolegy. Převážně s inženýry jiných oborů, ale také s managmentem a osobami zodpovědnými za provádění zkoušek. Popis testeru a data z testu jsem uváděl do protokolu, kde byly následně vyhodnoceny nezávislou zodpovědnou osobou.

2 Zadané úkoly

2.1 Test senzoru průtoku Huba Control 210

Obrázek 2.1 Vortexový průtokoměr Huba Control 210 [2]

2.1.1 Zadání

Hlavním cílem testu těchto senzorů bylo ověření funkčnosti a stability měření veličin při tom, když jimi protékalo znečištěné médium. Senzory bylo třeba ověřit před použitím v prototypu. Mým úkolem bylo zkonstruovat tester, který bude v dlouhodobém horizontu zaznamenávat data ze sedmi těchto senzorů, ukládat je pro pozdější analýzu a vytvářet základní vizualizaci.

Média, která protékala těmito senzory obsahovala různé emulze a špony kovů z brusného a řezného procesu. Celkem byly využity dvě směsi a jedna referenční tekutina, které byly pomocí pump cirkulovány skrze senzory. Tyto směsi byly vytvořeny tak, aby odpovídaly reálnému znečištění, kterému by byly senzory v zařízení vystaveny. Pro dvě znečištěné směsi byly použity tři za sebou řazené senzory a jedním referenčním senzorem protékala pouze neznečištěná kapalina, tedy kohoutková voda.

Dalším požadavkem bylo, aby byla snímaná teplota v samotných nádobách a bylo poté možno porovnat tuto hodnotu s teplotou změřenou průtokovými snímači.

2.1.2 Popis senzoru

Senzory typu 210 jsou založeny na využití principu Kármánovy vírové stezky. Vírové průtokoměry patří do skupiny rychlostních průtokoměrů, které vyhodnocují objemový průtok

na základě měření rychlosti proudícího média při znalosti průtočného průřezu. Pro měření rychlosti se využívá měření frekvence víru, které vznikají při obtékání překážky vložené do proudící tekutiny. Frekvenci víru lze měřit za pomoci senzoru tlaku, kdy frekvence výstupního tlaku je stejná jako frekvence vznikajících vírů. [3]

Řada senzorů 210 se skládá z mnoha různých variant, ať už s různým napájecím napětím, různým výstupním signálem nebo širokou škálou měřicích rozsahů. Jednotlivé typy mají různou jmenovitou světlost potrubí, která omezuje maximální průtok tekutiny a tím i měřicí rozsah. Senzor neobsahuje žádné pohyblivé části, a proto by neměl být citlivý na nečistoty. Má minimální tlakovou ztrátu a vysokou přesnost měření. Výstup senzoru je buď napěťový, proudový, pulzní nebo frekvenční.

Společně s měřením průtoku senzor poskytuje informaci o teplotě média. Měřicí element není ovšem přímo v kontaktu s měřeným médiem. Samotný měřicí element je termistor typu PT1000 s měřicím rozsahem -40 až +125 °C. V senzoru je poté již integrovaná jednotka pro zpracování signálu z termistoru nebo u některých typů je tento signál přímo vyveden na konektor. Výstup je realizován pomocí M12 konektoru s třemi nebo pěti vodiči. Třívodičová varianta neposkytuje možnost měření teploty. [4]

K testu byl určen senzor s jmenovitou světlostí DN8 s analogovým napěťovým výstupem 0-10 VDC pro snímání průtoku i teploty. Měřicí rozsah průtokoměru je 0,9 ÷ 15 $l \cdot min^{-1}$. Senzor potřebuje ke své funkci napájení, a to zdrojem se stejnosměrným napětím 11,5 až 33 V.

Obrázek 2.2 Tester průtokových senzorů

2.1.3 **Rozhraní pro řízení testu a konstrukce testeru**

Jako platforma pro řízení byl zvolen jednodeskový počítač Raspberry Pi 4 Model B. Tato platforma byla vybrána hlavně kvůli své velké flexibilitě, kdy je možné za poměrně krátký čas sestrojit komplexní zařízení dle požadavků. Zařízení disponuje 40 univerzálními vstupy/výstupy, dokáže s periferiemi komunikovat skrze rozhraní jako SPI, I2C, UART, 1-Wire atd. Operační systém platformy je Raspberry Pi OS, který vychází z filozofie UNIX systému. Na počítači lze pro tvorbu programu využít programovací jazyky jako je Python, JavaScript, C, C++, Ruby a mnoho dalších.

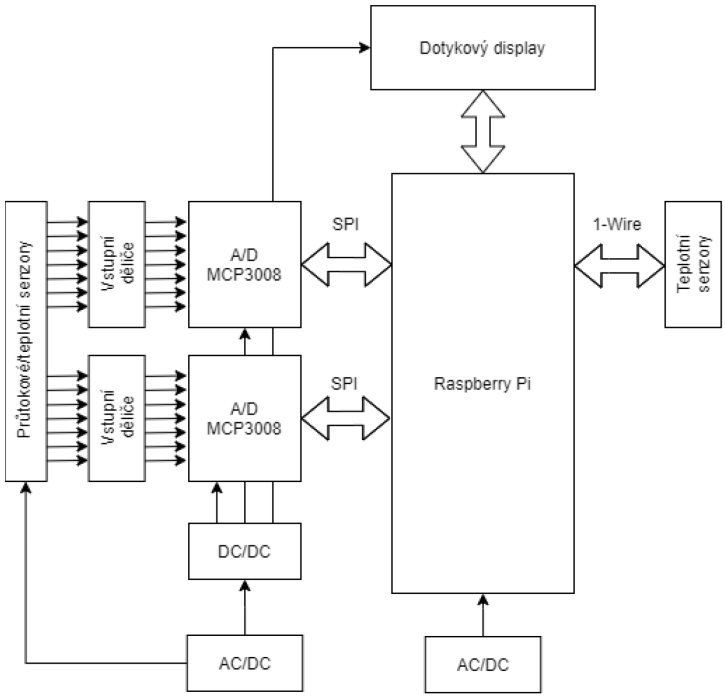
Vzhledem k tomu, že vstupní piny Raspberry Pi jsou pouze digitální a je potřeba snímat analogovou hodnotu, bylo nutné použít převodníky, které budou pomoci sběrnice komunikovat s ARM procesorem. Jako AD převodník byl zvolen převodník s postupnou aproximací MCP3008 firmy Microchip. Tento 10bitový převodník má 8 analogových vstupů s vstupním napětím 0-5 V za předpokladu, že převodník pracuje v single-ended módu a je připojeno 5V referenční napětí. Komunikace s převodníkem je řešena pomocí sériové sběrnice. Výstupní signály senzoru byly upraveny pomocí napěťového děliče pro přizpůsobení napěťových úrovní. Vzhledem k tomu, že je potřeba snímat 7x2 analogových signálů, bylo třeba v zapojení využít dva osmivstupové převodníky.

Propojení jednotlivých integrovaných obvodů a veškerých součástek bylo realizováno na univerzálním plošném spoji.

Pro vizualizace byl využit kapacitní dotykový 7palcový displej, který společně s Raspberry Pi a osazenou deskou plošných spojů byl umístěn do kompaktního pouzdra, a to později připevněno na konstrukci tvořenou hliníkovými profily. Pomocí průchodek jsou do tohoto pouzdra zavedeny veškeré kabely, ať už propojovací k senzorům, nebo napájecí vodiče pro samotný počítač.

Mechanická konstrukce není nijak složitá. Hlavní částí je L profil, na kterém jsou pomocí montážních úchytek připevněny hadice, mezi kterými jsou umístěny senzory. Tato část je dále připojena na hliníkové vzpěrky. Nádoby pro kapaliny jsou z plastu a horní kryt je vyroben z nerezové oceli. V krytu je otvor pro pumpu a rychlospojka pro připojení hadice, která zakončuje okruh cirkulace kapaliny. Pumpy jsou připojeny skrze hlavní vypínač. Zapojení asynchronních motorů pump je do hvězdy.

Program pro záznam a zpracování dat byl vytvořen za pomoci programovacího jazyku Python.



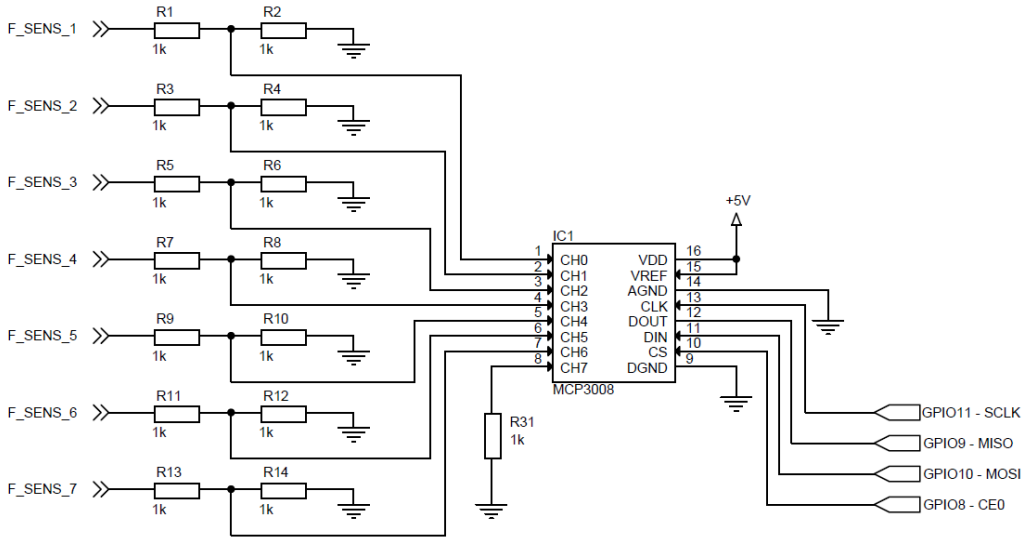
Obrázek 2.3 Blokové schéma jednotky pro záznam dat

Na blokovém schématu lze vidět propojení jednotlivých členů. Velká část komunikace mezi periferiemi a mikropočítačem probíhá skrze sběrnice.

2.1.4 **Návrh obvodu**

Vzhledem k absenci AD převodníku na desce počítače Raspberry Pi bylo nutné zvolit externí převodník. Byl zvolen převodník MCP300x od firmy Microchip. Jedná se o převodník, který se vyrábí ve 4 variantách. Konkrétně s označením MCP3001, MCP30002, MCP3004, a MCP3008. Poslední číslice v označení značí počet analogových vstupů. Vzhledem k tomu, že je nutné snímat 14 analogových veličin, tak byla zvolena varianta MCP3008 a ta byla použita dvakrát. Převodník komunikuje skrze sériovou sběrnici za pomocí SPI protokolu. Využití sériové linky usnadní připojení dvou převodníků k Raspberry Pi.

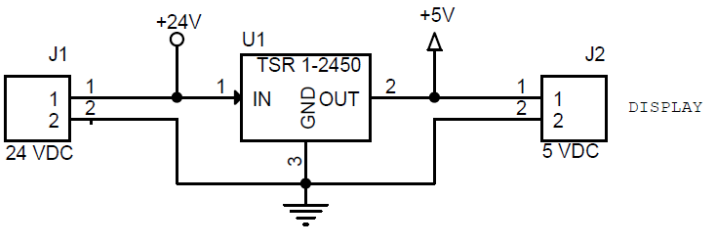
Na následujícím obrázku je schéma zapojení jednoho převodníku. V tomto případě se jedná o jeden ze dvou převodníků, konkrétně ten, který zpracovává signál popisující průtok. Jsou zde již zmíněné vstupní děliče, které ze signálu, který nabývá napětí 0-10 VDC, upravují na signál s napětím 0-5 VDC. Dělicí poměr je tedy 2:1. Signály jsou poté připojeny na vstupy MCP3008 a nevyužitý vstup je přes rezistor 1k uzemněn. Tímto způsobem bude zajištěna stabilní hodnota na 7. kanálu a nebude se jednat o vstup s plovoucím napětím. Piny CS, DIN, DOUT a CLK slouží k řízení a komunikaci po sériové sběrnici.



Obrázek 2.4 Zapojení AD převodníku

Napájecí napětí MCP3008 je od 2,7 VDC do 5,5 VDC. Volba napájecího napětí ovlivňuje i maximální možné vstupní napětí na analogových vstupech. V tomto případě je převodník napájen 5 VDC ze spínaného snižujícího měniče TSR 1-2450. Tento měnič je velice kompaktní a svým provedením lze zaměnit za lineární stabilizátor napětí řady 7805. Oproti lineárnímu stabilizátoru má ovšem efektivitu 84 % až 94 % a má menší rozkmit výstupního napětí. Při plném zatížení 1A je výstupní napětí udržováno na hodnotě 5 VDC s rozkmitem maximálně ± 2%. Další

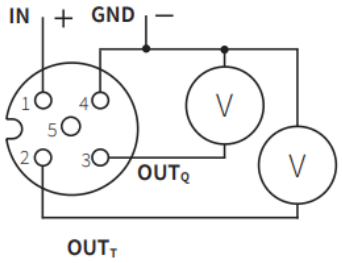
velkou výhodou je, že není nutno zapojovat kondenzátory na vstupní či výstupní svorky regulátoru. Výrobce doporučuje připojení vstupního kondenzátoru 22 µF / 50 V až při překročení 32VDC napájecího napětí. [5] Ze snižujícího měniče jsou napájeny dva převodníky a displej. Proud odebíraný z měniče je menší než hodnota maximálního nepřetržitého proudu dovolená výrobcem. Jeden převodník maximálně odebírá 0,5 mA a displej 0,5 A. Samotné Raspberry Pi je napájeno externím zdrojem. Tato separace je z toho důvodu, že při vypnutí napájení senzoru a displeje je stále možné se pomocí vzdálené plochy připojit k Raspberry Pi a pracovat s ním, ať už z důvodu přenosu dat, nebo úpravě softwaru.



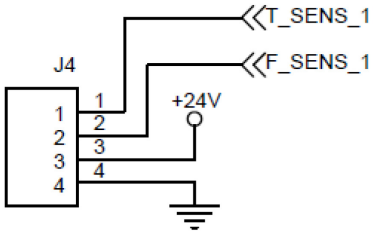
Obrázek 2.5 Zapojení snižujícího měniče TSR 1-2450

Zdroj 24 VDC je síťový spínání zdroj firmy Meanwell, ten je společně s asynchronními motory připojen skrze hlavní vypínač k síti.

Testované senzory jsou napájeny 24 VDC a poskytují dva analogové napěťové signály, které po úpravě vstupují do převodníku. Každý senzor je zapojen čtyřvodičově, kdy dva vodiče jsou použity pro napájení a další dva slouží k přenosu signálu.



Obrázek 2.6 Napěťový výstup senzoru [2]



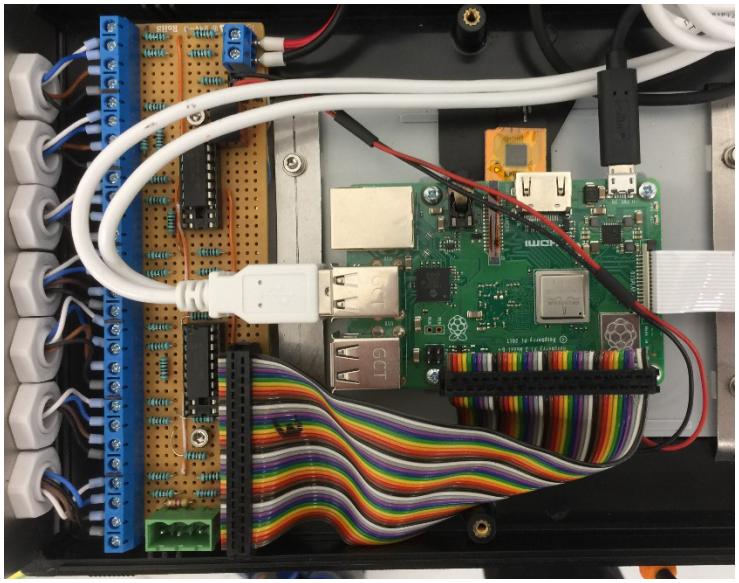
Obrázek 2.7 Zapojení senzoru na desce plošných spojů

K detekci teploty kapaliny v nádrži byla využita tři teplotní čidla DS18B20. Jedná se digitální snímač s rozlišením 9 až 12 bitů, přesnost lze nastavit zápisem do registru v paměti senzoru. Výchozí nastavení senzoru je s 12bitovou přesností, tato přesnost byla ponechána i při

provádění měření. Komunikace je skrze sériový protokol 1-Wire. Název protokolu zvýrazňuje jeho největší přednost, a to komunikace po jednom vodiči. Umožňuje i napájení senzoru za pomocí parazitního napájení, kdy je senzor připojen pouze 2 vodiči, a to datovým a zemí. Běžně se ovšem připojuje senzor třívodičově, tedy za pomocí dvou vodičů napájecích a jednoho datového. Každé zařízení připojené ke sběrnici má výrobcem nastavené neměnné unikátní 64bitové ID, které slouží jako adresa pro komunikaci. Při realizaci zapojení je nutné na datovou linku připojit pull-up rezistor, například s odporem 4,7 kΩ. [6] Sběrnice je vyvedena z hlavní řídicí jednotky do rozbočovače, který umožní připojit až 4 senzory ke sběrnici. Spojení je realizováno pomocí průmyslového konektoru se stupněm krytí IP68.

Obrázek 2.8 Rozbočovač pro 1-Wire

Obrázek 2.9 Senzor DS18B20



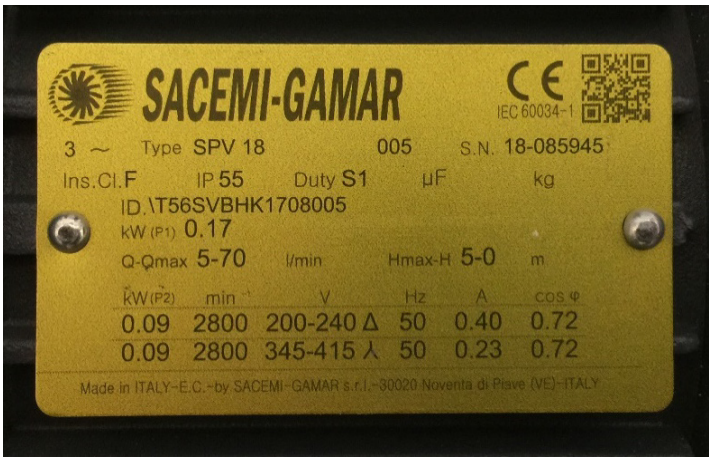
Obrázek 2.10 Vnitřní zapojení řídicí jednotky

Celé schéma zapojení obvodu pro převod dat a komunikaci s Raspberry Pi je dostupné v příloze.

2.1.5 **Pumpy k cirkulaci tekutin**

Pumpy Sacemi Gamar SPV18 jsou určeny pro přepravu kapalin obsahujících nečistoty do velikosti 3 mm. Pohon pumpy tvoří asynchronní motor v její horní části. Při testu byla využita trojice těchto pump.

Ze štítkových údajů lze vyčíst, že se jedná o 3fázový asynchronní motor s příkonem 0,17 kW. Druh zatížení je S1, motor je tedy konstruován pro trvalý chod. Při zapojení do hvězdy je poté výkon 0,09 kW při jmenovitých otáčkách 2800 min^{-1}. V síti 400/230 VAC 50 Hz lze tento motor přímo připojit na síť pouze v konfiguraci hvězda. Motory jsou skrze hlavní 3fázový vypínač a vidlici připojeny přímo do zásuvkové rozvodnice.



Obrázek 2.11 Štítek pumpy Sacemi-Gamar SPV 18

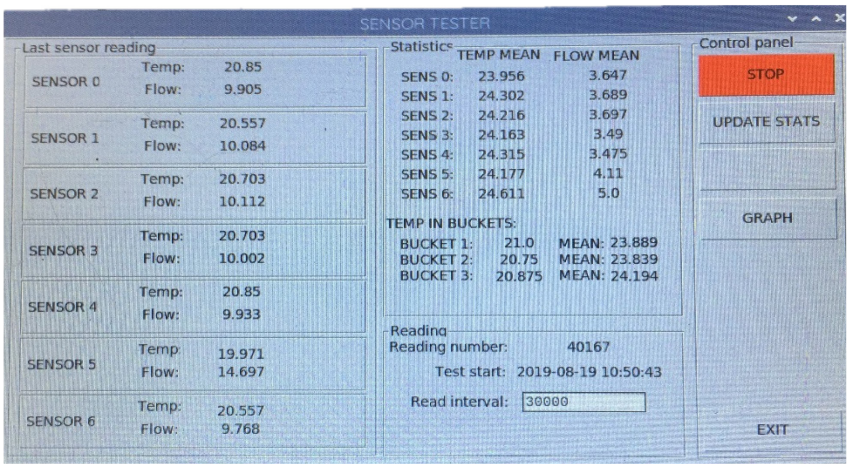
Obrázek 2.12 Pumpa Sacemi-Gamar SPV18 [7]

2.1.6 **Program pro záznam dat a vizualizaci**

Aplikace byla naprogramována v jazyce Python. Vzhledem k rozsáhlosti kódu jej nebudu podrobněji popisovat. Zaměřím se pouze na hlavní funkce.

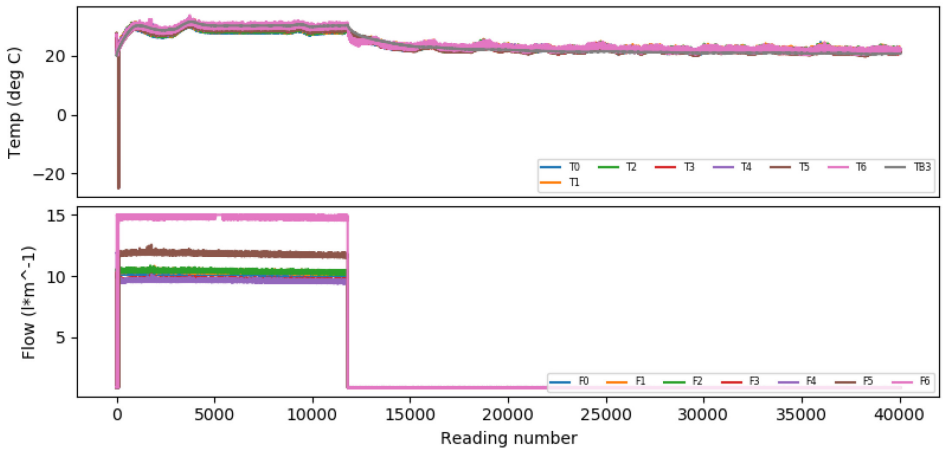
Nejdůležitější funkcí je čtení hodnot ze senzoru s nastavitelnou vzorkovací frekvencí a ukládání těchto dat do souboru, který lze později zpracovat v programu, jako je například MS Excel. Soubor s daty je ve formátu CSV. Jedná se tedy pouze o hodnoty oddělené separátorem. Obsahují vždy číslo datového vzorku, časové razítko, hodnotu průtoku a teploty z každého senzoru a teplotu média v nádobách.

Grafické rozhraní je plně ovladatelné pomocí několika tlačítek a textových boxů. V programu bylo využito několik knihoven. Jedna se hlavně o knihovny pro tvorbu grafického rozhraní či zpracování grafu. K tvorbě vizualizace, tedy GUI byla použita knihovna Tkinter. Pro zpracování dat do grafu byla využita knihovna Matplotlib.



Obrázek 2.13 Obrazovka vizualizace a nastavení testu

Data lze ve vizualizaci zobrazit po stisknutí tlačítka „GRAPH“. Výsledkem je náhledový průběh hodnot. Tento graf ovšem nesloužil pro přesnou analýzu. Vodorovná osa je pro oba průběhy totožná, vždy popisuje konkrétní číslo vzorku. Datové vzorky byly odebírány pro průtok i teplotu ve stejném intervalu, tedy vzorkovací periodě. Na svislé ose je průtok nebo teplota. Je zaznamenávána i teplota v nádržích, ta je poté označena jako TBx, kde x je identifikační číslo.



Obrázek 2.14 Grafické zobrazení změřených dat ve vizualizaci

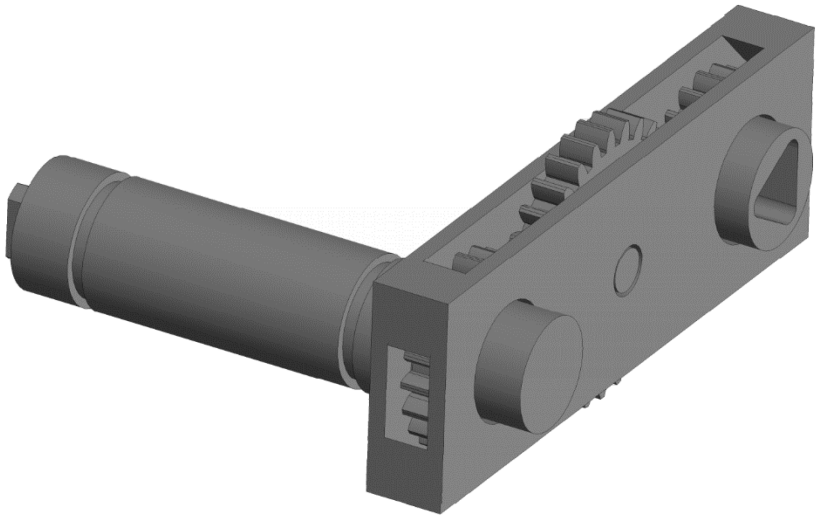
Na grafu je viditelné, kdy došlo k odpojení čerpadel. V čase přibližně 12 000 dojde k poklesu průtoku a také začne postupné snižování teploty kapalin. Hlavním důvodem tvorby těchto grafů bylo to, aby při průběžné kontrole testu byly patrné například zásadní změny hodnot. Zda například nedošlo ke kompletnímu selhání senzoru, zamezení průtoku v hadicích nebo chybě ze strany samotného zapojení.

2.2 Test 3D tištěné převodovky

2.2.1 Zadání

Cílem testu bylo ověření životnosti 3D tištěného komponentu. Jedná se o soustavu ozubených kol, která poskytuje možnost odjistit bezpečnostní spínač mimo samotné tělo. Vzhledem ke konstrukci zařízení, kde bude tento spínač využit, je nutno vyvést zdířku zámku na jinou pozici, kde je dostupná k zasunutí bezpečnostního klíče. Po zasunutí klíče a pootočení dojde k přenesení krouticího momentu na zámek samotného spínače, který lze tímto odjistit. Odjištění se používá například při nutnosti údržby nebo seřizování na stroji. Tento mechanismus musí být ověřen, aby byla zaručena životnost a funkčnost komponenty.

Mým úkolem bylo sestavit tester pro trojici těchto převodovek a zámku společně s jednoduchým řízením, které jsem měl za úkol také naprogramovat.



Obrázek 2.15 Model převodovky

2.2.2 Rozhraní pro řízení testu a konstrukce testeru

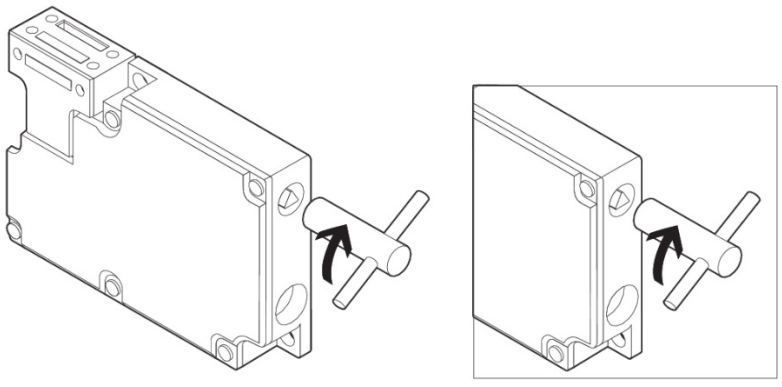
Vzhledem k nutnosti rotace klíče v zámku bylo nutné zvolit vhodné pohony. Pohony tvoří tři dvoufázové bipolární krokové motory od firmy JVL. Jedná se konkrétně o typ MST176A163C3AA30. Krokové motory byly zvoleny hlavně kvůli jednoduché řiditelnosti za pomoci modulu TMCM, který obsahuje integrované budiče. Nebylo tedy nutné použít další obvody a tím komplikovat zapojení. Tyto motory disponují dostatečným krouticím momentem k zamknutí a odemknutí mechanismu. Vysoký krouticí moment je dosažitelný při nízkých otáčkách. Na hřídelích motorů jsou upevněny koncovky, které přímo zapadají do zámku převodovky. Odpovídají konstrukci trojhranného klíče M5.

Pro řízení byl zvolen modul TMCM-3110 firmy Trinamic. K modulu lze připojit až 3 krokové motory s maximálním proudem pro jednu cívku až 2.8 A. Modul disponuje mnoha vstupy a výstupy a komunikačními rozhraními. Tento modul je podrobněji popsán v následující kapitole.

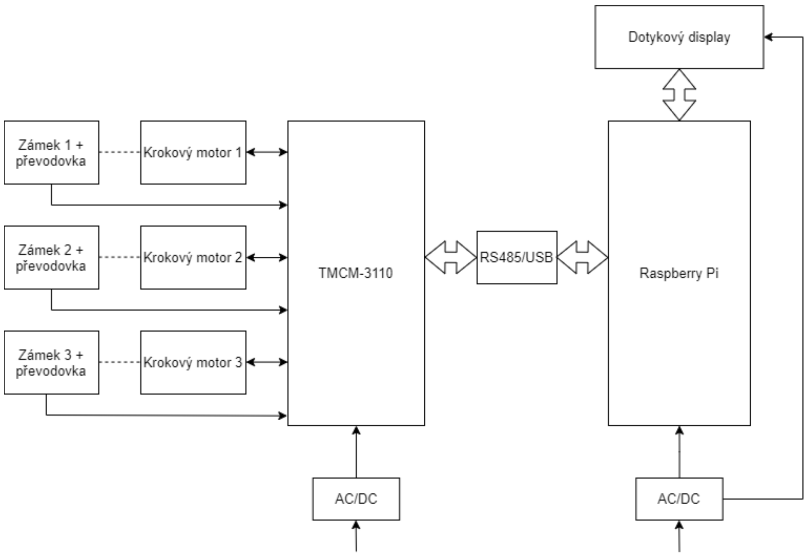
Dosažení pozice zamknutí a odemknutí je detekováno ze zpětné napěťové a proudové vazby motoru. Kdy při vzrůstu momentu zátěže na hřídeli dochází k posunu magnetického pole rotoru vůči statoru (zvětšení zatěžovacího úhlu) a zmenšení indukovaného napětí, které lze snímat a na jeho základě usoudit zatěžovací úhel, a tedy i zatížení motoru. Nutné je znát i proud tekoucí do cívek motorů. Se zmenšujícím zpětným indukovaným napětím na cívce motoru dochází k zvýšení proudového odběru. Z těchto údajů je modulem vypočtena hodnota StallGuard, která je využita pro zjištění koncových pozic. Zámek má dorazy v obou svých pozicích, které lze pomocí této metody detekovat a poté nemusí být použity koncové spínače.

Skrze RS485 komunikuje TMCM-3110 modul s RaspberryPi, díky kterému lze data o probíhajícím testu vizualizovat a dokáže test taky pozastavit. Mezi sběrnici USB od mikropočítače a sériovou linku modulu je vložen převodník. Pro vizualizace byl využit kapacitní dotykový 7palcový displej. Při běhu aplikace je také vytvářen jednoduchý log události, z kterého lze vyčíst data v případě chyby paměti na modulu. Vzhledem k nativní absenci rozhraní RS485 u RaspberryPi je připojen mezi modul a mikropočítač převodník RS485/USB.

Tři bezpečnostní spínače typu AZM 161SK-12/12RK0-024 byly použity při tomto testu. Opět se jedná o komponenty, které budou použity v prototypu. V zařízení je tento prvek použit kvůli bezpečnosti, kdy se eliminují rizika spojená s používáním. Spínač obsahuje několik kontaktů spínacích a rozpínacích, pomocí nichž lze vyhodnotit, zda je zasunuta vidlice, odjištěný zámek či další jiné stavy. Vidlice lze pomoci aktivace solenoidu uzamknout a tím zabránit otevření. Tyto funkce ovšem při testu nejsou využity. Solenoid je odpojen a detekuje se pouze úspěšné odjištění nebo zajištění pomocí rozpínacího kontaktu, který je mechanicky svázán s odjištěním. Tento signál je poté přenášen na modul TMCM, kde je možné pomoci něj stanovit počet spínacích cyklů.

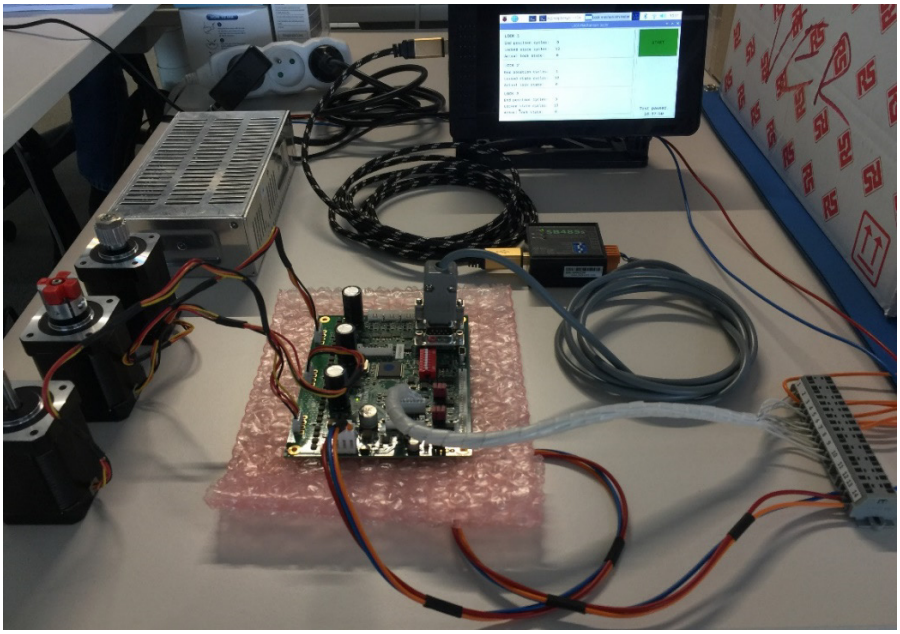


Obrázek 2.16 Bezpečnostní spínač – princip odblokování klíčem [8]

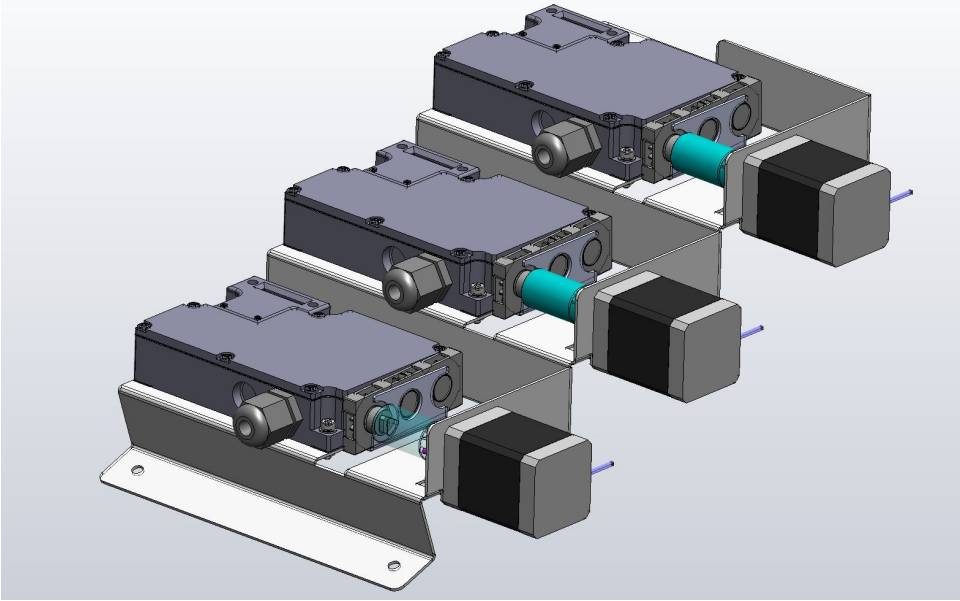


Obrázek 2.17 Blokové schéma testeru tištěných převodovek

V době tvorby této zprávy o absolvování odborné praxe bohužel ještě nebyla mým kolegou plně navržena a sestavena mechanická část testeru. Proto nemohu poskytnout fotky kompletního sestaveného zařízení. Veškeré zapojení i s programem je ovšem odzkoušeno a připraveno pro připevnění na konstrukci. Program je také plně funkční, jedinou nutnou budoucí změnou bude nastavení parametrů StallGuard, které je třeba přizpůsobit pracovnímu mechanismu.



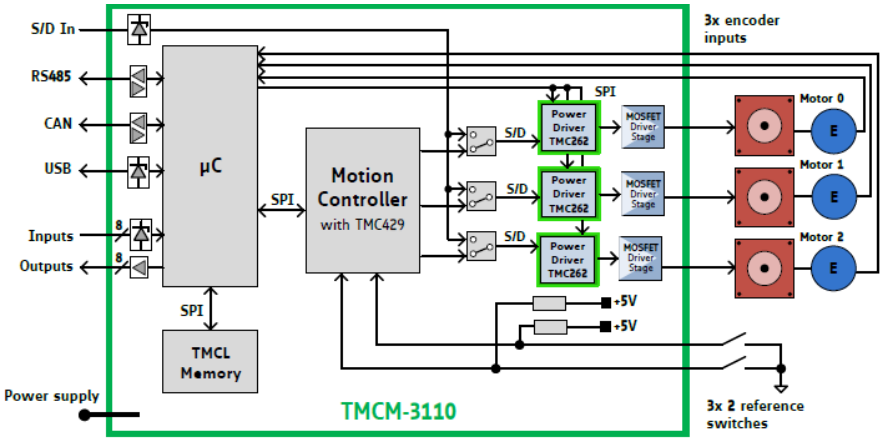
Obrázek 2.18 Zapojení pro odzkoušení funkce



Obrázek 2.19 3D model mechanismu testeru

2.2.3 Řídicí modul Trinamic TMCM-3110

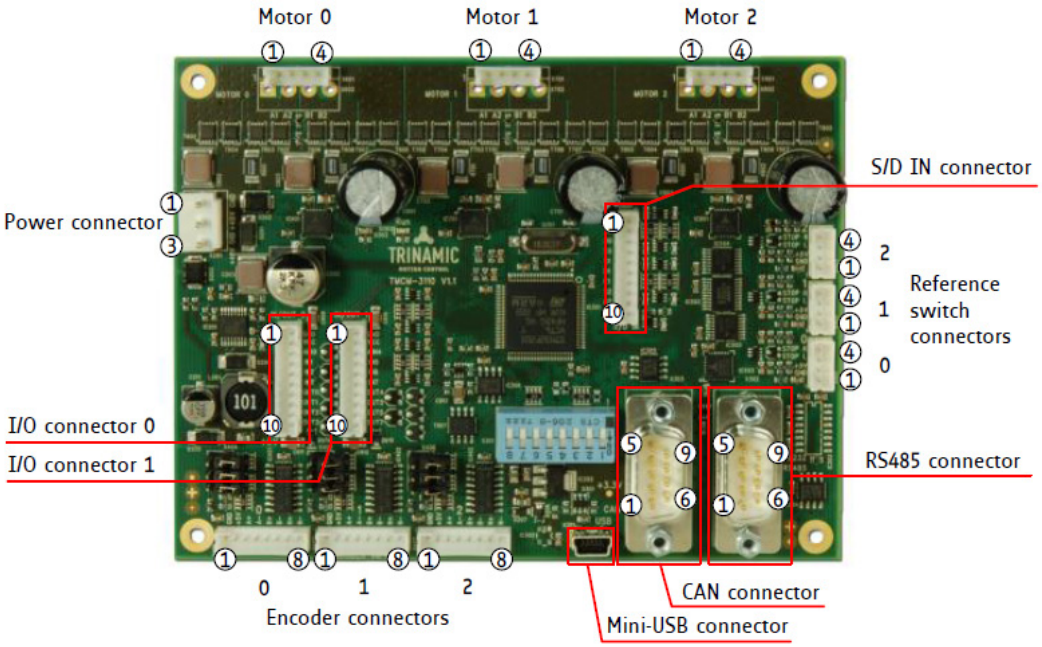
Nejdůležitější operace pro řízení testu vykonává tento modul. Jedná se o kompaktní 3osý budič krokových motorů s kontrolérem. Platforma je schopna řídit až 3 bipolární krokové motory s odběrem 2.8 A na vinutí a podporuje napájení až do 48 VDC. Ke každému motoru lze připojit referenční spínače a také inkrementační enkodéry. Komunikace je zajištěna skrze rozhraní jako je RS485, CAN nebo USB. Modul poskytuje 8 vstupů a 8 výstupů. [9]



Obrázek 2.20 Zjednodušené blokové schéma TMCM-3110 [10]

Díky firmwaru lze vykonávat program v samostatném (standalone) módu případně v přímém (direct) módu. Tyto módy lze také kombinovat, a pak může běžet program na kontroléru přímo integrovaném na desce, a přitom reagovat na příkazy z vnějšího systému. Hlavní program je implementován na modulu TMCM. Tento hlavní program řídí motory, kontroluje jejich zatížení a také zpracovává vstupní signály z bezpečnostních spínačů. Hodnoty jsou ukládany do paměti, kdy pomocí přímých příkazů posílaných z RPi je možné se na ně dotázat a mikrokontroler odpovídá na tyto žádosti. Přitom nepřerušuje chod vlastní smyčky. Dle náběžné hrany signálu spínače modulu zámku je inkrementována hodnota provedených zamykacích cyklů.

Společně s touto hodnotou je zaznamenávána hodnota dosažení koncových poloh zámku. Výrobce modulu poskytuje funkci tzv. StallGuard, která detekuje zatížení motoru a po překročení nastaveného limitu může dojít k přerušení. Pomocí tohoto přerušení je možné detekovat tyto koncové pozice.



Obrázek 2.21 Modul TMCM-3110 [9]

TMCM modul dále umožňuje například nastavování parametru jednotlivých pohonů v reálném čase, dynamické řízení proudu, funkce StallGuard a CoolStep a také obsahuje množství ochran. Deska má separované napájení pro řídicí obvody a pro napájení budičů motorů. Vstupní a výstupní obvody jsou kompatibilní s 24V napěťovými signály. Modul obsahuje také jeden 12bitový analogový vstup s rozsahem 0÷10 VDC. [9]

Platformu lze řídit přímými příkazy, nebo může samostatně vykonávat program uložený v paměti. Program se zapisuje pomocí jazyku TMCL. Instrukce tohoto jazyka jsou velice podobné

jazyku symbolických adres. Zápis kódu je možný v TMCL-IDE, kde lze využívat mnemonický formát jednotlivých instrukcí. Instrukce se poté překládají do jejich binární reprezentace a lze je nahrát do modulu a spustit. Instrukce jsou děleny do několika kategorií. Jedná se například o instrukce, které jsou využívány pro řízení motoru, přesun dat, volání podprogramů, výpočty nebo nastavení přerušení.

Díky velkému množství funkcionalit je tento modul velice flexibilní a lze ho využít pro mnoho aplikací. Primárně je určen pro řízení 3-osých systému.

2.2.4 **Práce s pamětí modulu**

Pro práci s mikrokontrolerem na modulu je velice důležité znát strukturu paměti. Výrobce přesně definoval využití jednotlivých bank v paměti. Veškeré parametry jsou rozdělené do 4 bank. Banky obsahují globální parametry, s kterými lze manipulovat pomocí instrukcí SGP, GGP, AGP, STGP a RSGP. Instrukce SGP slouží k nastavení parametru na konstantní hodnotu, využitelná například při inicializaci. Instrukce GGP se používá pro přečtení hodnoty na adrese v paměti a zkopírování obsahu do akumulátoru pro další použití. AGP je určena pro zkopírování obsahu v akumulátoru do proměnné, tedy do jiného globálního parametru. Dvojice instrukcí STGP a RSGP slouží pro ukládání a obnovy dat z EEPROM.

Banka 0 obsahuje parametry, kterými lze konfigurovat vlastnosti modulu. Mezi ně patří například adresa modulu, modulační rychlost sériové sběrnice RS485, aktuální hodnota programového čitače a mnoho dalších.

Banka 1 není běžně uživateli dostupná.

Do banky 2 můžeme uložit až 256 proměnných o velikosti 32 bitů. Prvních 56 proměnných lze uložit i do paměti EEPROM a tím zamezit ztrátě dat při přerušení napájení. Proměnné mohou nabývat i záporného znaménka a mají rozsah -2 147 483 648 až 2 147 483 648.

V poslední bance nalezneme parametry pro nastavení přerušení. Lze zde nastavit vlastnosti časovačů a čítačů, spínací úrovně externích přerušení atd. [10]

2.2.5 **Program pro TMCM-3110**

K naprogramování modulu byl využit jazyk TMCL. Jedná se o jazyk, který výrobce doporučuje. Strukturou se velice podobá jazyku symbolických adres. Vzhledem k předchozí zkušenosti s jazykem symbolických adres nebylo složité pochopit princip jazyka TMCL. Výrobce dodává kompletní podklady pro tento jazyk i s detailním popisem instrukcí. Vzhledem k délce kódu zde uvádím jen jeho nejdůležitější části pro popis funkce.

V prvé řadě bylo nutno vytvořit zadní program. V tomto zadním programu dochází k vstupování do podprogramů (pomocí instrukce CSUB) a načítání globálních parametrů do akumulátoru (instrukce GGP). Tato část programu se cyklicky opakuje, vzhledem k nepodmíněnému skoku na konci. Nepodmíněný skok je proveden instrukcí JA, kdy její první operand označuje návěští loop. Na toto návěští je skok přesměrován. Je možné si všimnout tří bloků, kdy jeden blok je vždy pro obsluhu jednoho motoru. Algoritmus řízení jednotlivých motorů se neliší, pracuje pouze s jinými parametry.

```
// *** Main Loop *** //
loop:
    GGP m0_direction, 2         // Copy content of the var.to accu
    CSUB m0_dirCheckRight       // Call subroutine dirCheckRight
    CSUB m0_dirCheckLeft        // Call subroutine dirCheckLeft

    GGP m1_direction, 2         // Copy content of the var.to accu
    CSUB m1_dirCheckRight       // Call subroutine dirCheckRight
    CSUB m1_dirCheckLeft        // Call subroutine dirCheckLeft

    GGP m2_direction, 2         // Copy content of the var.to accu
    CSUB m2_dirCheckRight       // Call subroutine dirCheckRight
    CSUB m2_dirCheckLeft        // Call subroutine dirCheckLeft

    JA loop                     // Jump to loop
```

Obrázek 2.22 Ukázka kódu, zadní program

Pro lepší orientaci v kódu se využívají symbolické konstanty. Poté lze k jménu proměnné přiřadit konstantu, například adresu, na které se v paměti bude hodnota uchovávat. Následně pro čtení a zápis nemusíme znát adresu v paměti, ale stačí znát pouze název proměnné. Zápis kódu je poté následující.

```
// *** Symbolic names for variables *** //
m0_direction = 10
m0_count = 11

// *** Variable initialization *** //
SGP m0_direction, 2, 0              // Set val at addr of direct. to 0
```

Obrázek 2.23 Ukázka kódu, symbolické konstanty

Je zde viditelné i využití tohoto principu u instrukce SGP, která slouží k zápisu globálního parametru. První operand označuje index v bance, druhý operand číslo banky a třetí operand je samotná hodnota.

Pro nastavení základních parametrů pro buzení motoru je využito instrukcí SAP, tedy nastavení parametru osy. Touto instrukcí lze nastavit například maximální rychlost pozicování, aktuální pozici, maximální zrychlení a je zde možnost i řídit proud, který budič motoru dodává.

```
// *** Initialization / Set up axis parameter *** //
// Motor 0
SAP 1, 0, 0                    // Reset actual position to 0
SAP 4, 0, vmax                 // Set max. positioning speed
SAP 5, 0, amax                 // Set max. acceleration
SAP 6, 0, cmax                 // Set max. current
```

Obrázek 2.24 Ukázka kódu, nastavení parametru osy (motoru)

Vzhledem k využití funkce označené jako StallGuard bylo nutné inicializovat parametry i pro ni. StallGuard slouží k detekci zatížení motoru. Při překročení limitů, které byly pro motor nastaveny lze vyvolat přerušení. Operandy instrukce se skládají z indexu parametru osy, indexu motoru (osy) a hodnoty.

```
// *** stallGuard™ initialization *** //
// Motor 0
SAP 173, 0, 1                  // Set stallGuard filter disable
SAP 174, 0, stallGuardValue    // StallGuard threshold
SAP 181, 0, 900                // Min. speed for motor to be stopped
```

Obrázek 2.25 Ukázka kódu, nastavení parametru funkce StallGuard

Při aktivaci výše zmíněné funkce dojde při určitých podmínkách k přerušení a vykonání přerušovací rutiny. Nejprve je nutné nastavit vektor přerušení a následně povolit přerušení. K nastavení přerušovacího vektoru se používá instrukce VECT, k povolení přerušení instrukce EI.

```
// *** Interrupt initialization *** //
// Motor 0
VECT 15, m0_stallRoutine       // Defines the interrupt vector for the
stallGuard - motor 0 interrupt
EI 15                          // Enable stall interrupt for motor 0
```

Obrázek 2.26 Ukázka kódu, nastavení přerušení

Přerušovací rutina vypadá takto. Po vykonání rutiny je potřeba vyvolat návrat z přerušení. K tomuto slouží instrukce RETI, která pomocí změny hodnoty v PC se navrátí za instrukci, která byla naposledy vykonána před vyvoláním přerušení.

```
// *** Interrupt routine stallGuard*** //
// *** M0 - stallGuard routine *** //
m0_stallRoutine:
    MST 0                       // Stop motor 0
    SAP 1, 0, 0                 // Set position as zero
    GGP m0_direction, 2         // Get global parameter
    CALC NOT, 0                 // Invert value in ACCU
    AGP m0_direction, 2         // Copy ACCU value to direct at bank 2
    CSUB m0_count               // Call Count subroutine
    RETI                        // End of interrupt
```

Obrázek 2.27 Ukázka kódu, rutina přerušení

K detekci impulzu ze samotného zámku se využívá vyvolané přerušení na digitálním vstupu modulu. Při reakci na náběžnou hranu dochází k vyvolání přerušení a je vykonána přerušovací rutina. Opět je nutné, aby byly nastaveny přerušovací vektory a povolené přerušení. Pomocí zápisu v paměti lze také zvolit, zda má dojít k přerušení v závislosti na náběžnou hranu, sestupnou hranu apod.

Mikrokontroler tedy cyklicky opakuje instrukce v hlavní smyčce, tj. mezi návěštím loop a nepodmíněným skokem. Mezi vykonáváním instrukcí kontroluje veškeré zdroje přerušení, ať už z digitálních vstupů či funkce StallGuard. Při přerušení dochází k vykonání rutiny a návratu do hlavní smyčky. Rutina přerušení StallGuard mění směr otáčení motorů a inkrementuje počítadlo, které obsahuje informaci o počtu dosažených koncových pozic. Při přerušení z digitálního vstupu dochází k inkrementaci jiného počítadla, které zaznamenává počet úspěšných zamknutí bezpečnostního zámku. Data uložená v počítadlech jsou zálohována do paměti EEPROM, která je energeticky nevolatilní, a tedy nedochází k ztrátě dat při výpadku napájení. Z těchto údajů lze poté vyhodnotit výsledky funkčního testu převodovky.

2.2.6 **Program komunikující s modulem**

Aby bylo možné data o probíhajícím testu srozumitelně interpretovat, bylo nutno vytvořit jednoduchou vizualizaci. K vizualizaci byl zvolen dotykový displej, na který lze umístit i ovládací prvky a pomocí nich řídit průběh testu. Displej je připojen k jednodeskovému mikropočítači Raspberry Pi 4, který s modulem TMCM komunikuje skrze sériovou linku. Sériová linka je následně pomocí převodníku připojena do USB portu Raspberry Pi.

Pomocí instrukcí z mikropočítače se lze dotazovat na stavy globálních parametrů v bankách, které se s chodem hlavního programu na modulu mění. Tyto proměnné popisují již dříve zmíněné stavy, jako je počet dosažení koncových pozic, či počtu úspěšných odjištění zámku. Mikrokontroler na modulu zpracovává i instrukce přicházející po sériové lince a adekvátně na ně odpovídá. Je ovšem nutné mít správně nastaveny komunikační parametry a mít tuto komunikaci povolenou.

Program byl zapsán v programovacím jazyce Python. Při tvorbě byl hojně využit modul PythonTMCL, který usnadňuje implementaci komunikace. Modul má již vytvořené funkce, které se starají o správný formát zprávy, výpočet kontrolního součtu (checksum) anebo zpracování odpovědi od modulu.

Před navázáním komunikace bylo nutné programově nastavit parametry komunikace a odkázat na port, kde je připojen převodník pro USB-Sériovou linku. Následně dochází k vytvoření objektů a přiřazení modulu a jednotlivých motorů.

```
# Serial-address as set on the TMCM module.
MODULE_ADDRESS = 1

# Open the serial port presented by your RS485 adapter
serial_port = Serial("/dev/ttyUSB0")

# Create a Bus instance using the open serial port
bus = pyTMCL.connect(serial_port)

## Get the module module with address MODULE_ADDRESS
module = bus.get_module(MODULE_ADDRESS)

motor0 = module.get_motor(0)
motor1 = module.get_motor(1)
motor2 = module.get_motor(2)
```

Obrázek 2.28 Ukázka kódu, parametry komunikace a inicializace motorů

Po tomto nastavení lze již s modulem komunikovat za použití instrukcí, které poskytuje výrobce. Instrukce je modulu zaslána příkazem „bus.send“. Objekt bus specifikuje konkrétní připojený sériový port. Například zde se jedná o příkaz, kdy je na modul zaslána instrukce, která vede k zastavení běhu samostatného programu na TMCM modulu. Odpověď modulu je uložena v objektu response.

```
response = bus.send(MODULE_ADDRESS, 128, 0, 0, 0) # Stop application
```

Obrázek 2.29 Ukázka kódu, příkaz pro odeslání instrukce do modulu

V odpovědi je zakomponováno několik údajů, jedná se o atributy. Jsou to například:

- adresa, kam je odpověď zasílána (cíl),
- adresa modulu,
- status,
- instrukce,
- hodnota,
- kontrolní výpočet (checksum).

Instrukce v odpovědi je specifikována pomocí čísla instrukce. Hodnota poté může obsahovat výslednou hodnotu pro provedení instrukce, zde se jedná o desetinné číslo. Status svou hodnotou popisuje, zda došlo k správnému vykonání instrukce nebo nikoliv. V případě chyby specifikuje konkrétní chybu. Takže například po úspěšné vykonání instrukce je status 100, v případě chybně zadané instrukce nabývá status hodnoty 2. Těchto hodnot je celkem 8.

Při zápisu instrukce před odesláním je nutné specifikovat operandy instrukce, ty se píšou za číslo instrukce a jsou odděleny čárkou. Význam operandů se liší dle konkrétní instrukce.

Zde je již část rutiny pro čtení proměnných z TMCM v 2. bance. Konkrétně z adres 11, 21, 31 a přiřazených hodnot do globálních proměnných v Pythonu. Rutina se cyklicky opakuje každých 10 ms. Časování rutin je řešeno pomocí příkazu z modulu Tkinter, který se stará o grafickou stránku vizualizace.

```
def tmcm_routine():
    global mot1InterruptCount, mot2InterruptCount, mot3InterruptCount
    # Get the value of the global parameter (command 10, memonic=GGP)
    # 2 = bank, 20 = adress (0-255)

    response = bus.send(MODULE_ADDRESS, 10, 11, 2, 0)
    mot1InterruptCount = response.value

    response = bus.send(MODULE_ADDRESS, 10, 21, 2, 0)
    mot2InterruptCount = response.value

    response = bus.send(MODULE_ADDRESS, 10, 31, 2, 0)
    mot3InterruptCount = response.value

    root.after(10, tmcm_routine)
```

Obrázek 2.30 Ukázka kódu, rutina pro čtení proměnných v 2. bance

Mezi další cyklicky opakované rutiny patří blok pro validaci koncových pozic. Při testu bylo bráno v úvahu, že v případě zablokování motoru může dojít k časté aktivaci funkcí StallGuard a zbytečnému zatěžování motoru. Proto se v nastavené periodě, konkrétně každých 5 sekund, kontroluje počet přerušení, jejichž zdrojem je StallGuard. Pokud algoritmus vyhodnotí, že došlo k častému sepnutí, pozastaví test a vyčkává na obnovení obsluhou. Při pozastavení testu dojde k zastavení všech motorů a výpisu chybové hlášky ve vizualizaci. Chyba je také s časovým razítkem zapsána do logu. Přikládám část kódu, která tuto funkci vykonává. Pro každý motor je tato funkce obdobná, tedy ve funkci validation_routine jsou zakomponovány bloky pro kontrolu všech 3 motorů.

```
def validation_routine():
    global previousInterruptCount
    global mot1InterruptCount

    if(mot1InterruptCount >= previousInterruptCount[0] + interruptMaxDif):
        print("ERROR on the lock 1, too many stallGuard interrupts")
        response = bus.send(MODULE_ADDRESS, 128, 0, 0, 0)
        motor0.stop()
        motor1.stop()
        motor2.stop()
        print("TMCM CMD send - STOP APPLICATION")
        log = open(log_name, "a")
        log.write("\n\nERROR: Lock 1 error, too many stallGuard interrupts: "
+ script_start_date +  "\n")
        log.close()
        lock1FrameNameLabel.config(text="LOCK1   ERROR - too many stallGuard
interrupts", fg="RED")
        lock1FrameNameLabel.config
        lock1FrameNameLabel.update()
        runState = False
    previousInterruptCount[2] = mot3InterruptCount
    root.after(4000, validation_routine)
```

Obrázek 2.31 Ukázka kódu, validační rutina

Další funkce v programu slouží pro vykreslení GUI, a kontroly prvků souvisejících s grafickým rozhraním. Dochází zde ke kontrole stisku tlačítek a aktualizaci dat na obrazovce. Celý kód zde opět neuvádím. Veškeré funkce, které dokážou manipulovat s grafickými prvky jsou použity z modulu s názvem Tkinter. Jako příklad vkládám část kódu, která se vykoná při stisknutí tlačítka STOP/START na dotykovém displeji se spuštěnou vizualizací. Tlačítko je určeno pro pozastavení či spuštění průběhu testu. Tlačítko je takto vytvořeno jako grafický prvek. Jednotlivými argumenty funkce jsou nastaveny jeho parametry, jako vzhled, text či funkce, která se volá při stisku.

Obrázek 2.32 Ukázka okna vizualizace testeru tištěných převodovek

```
stopBtn = Button(controlFrame, width=15, heigh=5, text="Start", bg="green",
font=("Courier", 14), command=stopBtn_click)
stopBtn.pack(side=TOP, pady=2, padx=2)


# Button callbacks
def stopBtn_click():
    global runState
    runState = not runState
    print("Test running:", runState)

    if runState == True:
        response = bus.send(MODULE_ADDRESS, 129, 1, 1, 0)
        motor0.stop()
        motor1.stop()
        motor2.stop()
        print("TMCM CMD send - STOP APPLICATION")
    else:
        response = bus.send(MODULE_ADDRESS, 128, 0, 0, 0)
        motor0.stop()
        motor1.stop()
        motor2.stop()
        print("TMCM CMD send - RUN APPLICATION")
```

Obrázek 2.33 Ukázka kódu, nastavení tlačítka jako grafického prvku a jeho funkce

Společně s během programu se periodicky zapisují data do logu, který je ukládán na paměťové kartě mikropočítače. V logu jsou údaje o spuštění testu, chybách a také veškerá data získávána z modulu. Jedná se opět o hodnoty, které jsou uloženy v bankách pamětí mikrokontroleru a skrze sériovou linku jsou přenášeny do Raspberry Pi.

3 Získané teoretické a praktické znalosti

V průběhu praxe jsem se naučil mnoho užitečných věcí. Za velice podstatné beru to, že jsem se naučil, jakým způsobem jsou vyvíjená zařízení určena k produkci. Dozvěděl jsem se například to, jak je vývoj rozdělen do několika fází, ve kterých se postupně ověřují jednotlivé koncepty a staví prototypy. Nečekal jsem, že při tvorbě prototypu a výběru komponent je kladen tak velký důraz na ověření životnosti, omezení poruchovosti a zajištění dlouhodobě správné funkce. Nyní mi tato priorita dává veliký smysl, vzhledem k tomu, že změna na prototypu je oproti změně v globální produkci časově a ekonomicky zanedbatelná. Celá strategie vedení projektu pro mě byla také nová. Zjistil jsem, jak důležitá je komunikace mezi jednotlivými členy týmu a také jakou podstatnou roli hraje vzájemná diskuse a konstruktivní kritika.

Vzhledem k aplikaci elektroniky jsem se naučil, jakým způsobem pracovat s AD převodníky a dalšími obvody. Nové znalosti jsem získal ohledně sběrnic a komunikačních protokolů, které využívají mikropočítače. Zdokonalil jsem si způsoby zpracování a ukládání dat a jejich vizualizaci. Naučil jsem se tvořit jednoduché aplikace za pomocí programovacího jazyka Python.

Seznámil jsem se také se způsoby řízení krokových motorů a detekce jejich zatížení. Osvojil jsem si základní zapojení krokových motorů a obvodů pro jejich řízení. Získal jsem znalosti o řídicích modulech TMCM firmy Trinamic a jejich programování v jazyku TMCL. Dokázal jsem zprovoznit komunikaci mezi dvěma systémy, a to Raspberry Pi a TMCM-3110.

Naučil jsem se i o přípravě vzorku v metalografickém procesu. Měl jsem možnost pracovat se zařízeními, které slouží k řezání nebo leštění vzorků. Bylo mi vysvětleno i několik důležitých parametrů, které je třeba dodržet, tak aby vzorek měl správnou kvalitu ke zkoumání.

Také jsem si poměrně zlepšil angličtinu, vzhledem k tomu, že ostravský vývojový tým často komunikuje s dánskými kolegy. V některých případech jsem musel o svých počinech vést diskusi a obhajovat své řešení a výsledky. Většina dokumentů, konkrétně datasheetu komponent, je také často v anglickém jazyce. Z těchto dokumentů jsem se naučil mnoho nových technických slovíček a frází.

4 Scházející znalosti

Scházely mi především praktické zkušenosti ze sestavování vlastních obvodů využívajících mikropočítače a jejich periferie. Při studiu jsem se setkal se základními obvody, ale ne například s tím, jak správně tyto obvody aplikovat společně s mikropočítačem. Teoreticky jsem znal mnoho informací, ale některé zásady při praktickém návrhu obvodu jsem postrádal.

5 Dosažené výsledky a závěr

V době odborné praxe ve firmě jsem navrhnul dva testery a mnoho malých dílčích úkolů. První tester sloužil k ověření funkčnosti a stability teplotních a průtokových snímačů v stanovených podmínkách. V tomto případě jsem navrhnul zapojení, naprogramoval řídicí systém a tester kompletně sestavil. Test proběhl. Při testování nenastal žádný technický problém a data byla zaslána odborníkovi na vyhodnocení a senzory byly poskytnuty na vizuální inspekci.

U testeru 3D tištěných převodovek jsem byl zodpovědný za návrh zapojení a naprogramování řídicího systému. Mechanickou konstrukci tvoří kolega a v době psaní této bakalářské práce se na ní stále pracuje. Nicméně mnou navržené zapojení je funkční a kompletně programově dokončeno. Před spuštěním testu očekávám odladění a odstranění chyb, které se objeví po odzkoušení s mechanikou.

Při absolvování praxe jsem dokázal aplikovat teoreticky získané znalosti z dosavadního studia. Došlo také k získání mnoha nových znalostí. Ať už se jedná o nově naučené základy programovacího jazyka TMCL, návrhu obvodu s AD převodníky či sběru dat ze senzoru. Získal jsem i nové praktické zkušenosti. Při práci jsem mohl pracovat s velmi moderním vybavením firmy a aplikovat nové technologie. Při návrhu obvodů jsem získal orientaci v technické dokumentaci elektronických komponentů a rozšířil jsem si povědomí o některých zapojeních. Řekl bych, že také došlo ke zlepšení mé technické angličtiny společně s její verbální formou. Měl jsem možnost komunikovat s kolegy ze zahraničí, kdy jsme komunikovali pouze v anglickém jazyce. Vzhledem k tomu, že firma navrhuje a vyvíjí zařízení pro materiálografii, tak jsem získal i nové informace z tohoto odvětví.

Použitá literatura

[1] *Structure 36, Struers Journal of Metalography: The History of Struers.* 2000.

[2] *Vortex flow sensor 210. In: Huba Control* [online]. Würenlos: Huba Control, 2020 [cit. 2020-11-23]. Dostupné z: https://www.hubacontrol.com/fileadmin/_processed_/3/6/csm_210_DurchflusssDurch_2_30d74396a8.png

[3] *Automa: vírové průtokoměry – princip, vlastnosti a použití* [online]. 2014. 2014 [cit. 2020-11-24]. ISSN 1210-9592. Dostupné z: http://automa.cz/Aton/FileRepository/pdf_articles/53030.pdf

[4] *Flow sensor for liquid media Type 210* [online katalogový list]. In: . Würenlos: Huba Control, 2020 [cit. 2020-11-24]. Dostupné z: https://www.hubacontrol.com/fileadmin/user_upload/domain1/Produkte/EN/Datenblatt/210_Flow_sensor.pdf

[5] *TSR 1 Datasheet* [online katalogový list]. In: . Traco Electronic, 2020, s. 1-3 [cit. 2020-11-24]. Dostupné z: https://tracopower.com/tsr1-datasheet/

[6] GUIDE TO 1-WIRE COMMUNICATION: TUTORIALS 1796. *Maxim Integrated* [web]. Maxim Integrated Products, 2008, 19 Jun, 2008 [cit. 2020-11-24]. Dostupné z: https://www.maximintegrated.com/en/design/technical-documents/tutorials/1/1796.html

[7] Electric pump SPV18. In: *SACEMI GAMAR* [online]. Via A. Pacinotti, Italy: Sacemi Gamar, [2021] [cit. 2021-02-24]. Dostupné z: https://www.sacemigamar.com/site/assets/files/1835/spv12_18.jpg

[8] Solenoid interlock AZM161SK-12/12RK-M16-24V. In: *Schmersal* [online]. K.A. Schmersal GmbH & Co. KG, Möddinghofe 3, D-42279 Wuppertal: Schmersal, 2021 [cit. 2021-02-24]. Dostupné z: https://products.schmersal.com/en_GB/product/934/azm161sk-1212rk-m16-24v

[9] *TMCM-3110 HARDWARE MANUAL: Hardware Version V1.11* [online]. In: . Hamburg, Německo: TRINAMIC Motion Control GmbH & Co., 2018, s. 1-32 [cit. 2020-11-24]. Dostupné z: https://www.trinamic.com/fileadmin/assets/Products/Modules_Documents/TMCM-3110_hardware_manual_Hw1.10_Rev1.04.pdf

[10] *TMCM-3110 TMCL Firmware Manual: Firmware Version V1.11 | Document Revision V1.05* [onlinel]. In: . Hamburg, Německo: TRINAMIC Motion Control GmbH & Co., 2017, s. 1-111 [cit. 2020-11-24]. Dostupné z: https://www.trinamic.com/fileadmin/assets/Products/Modules_Documents/TMCM-3110_TMCL-firmware_manual_Fw1.14_Rev1.11.pdf

Seznam příloh